IN THE SPECIFICATION

Please replace Paragraph [0002] on Page 1 in the section entitled "Background" with the following paragraph:

-- **[0002]** A high-speed link is a point-to-point interconnect that transfers data between two components using a link transfer protocol. Using high-speed differential signaling and sophisticated clocking, links are replacing buses as the main interconnect between different components (such as, a processor, a chipset, an input/output bridge, etc.) within a computer system. Links make use of a link transfer protocol that is different from a bus transfer protocol. For example, in the case of the link transfer protocol, transactions in links are broken up into requests and replies to increase scalability and to hide transfer latency. --

Please replace Paragraph [0006] on Page 3 in the section entitled "Background" with the following paragraph:

-- **[0006]** Structure based functional tests (SBFT) and functional random instruction tests for speed (FRITS) are execution-based test methodologies designed to address the ATE's speed and input/output bandwidth issues. Under SBFT's and FRITS' methodologies, a test code is first loaded into a DUT's internal storage, [[(]]for example, [[are]] the caches in a processor[[)]]. Thereafter, the test code is executed and is used to test different parts of the DUT. Because all testing is done internally, the ATE is effectively decoupled from component testing, thus solving the ATE's speed and I/O bandwidth and signaling problems. One drawback with this type of testing is that it

does not cover the DUT's input/output channels as well as the associated protocol, crossbar and link control layers. In order to extend SBFT and FRITS's type tests to cover a DUT's input/output channel and all these other associated logic, ATE's will be required to provide the proper IO responses at the right time. However, this approach would be limited by the ATE's speed, IO bandwidth and signaling complexity problems. - -

Please replace Paragraph [0014] on Page 6 in the section entitled "Detailed Description" with the following paragraph:

- - **[0014]** Referring now to Figure 2 of the drawings, a DUT [[100]] 200 is shown to include a central processing unit (CPU) core 202, a cache memory 204, and a front side bus 206. The DUT 200 also includes a test access port 208 which in one embodiment may be a Joint Test Action Group (JTAG) port as defined in the Institute of Electrical and Electronic Engineers (IEEE) 1149.1 specification. Also shown in Figure 2 is an ATE 210 which includes assembled or compiled test codes 212 stored in a memory area 214. In use, the assembled/compiled test codes 212 are input into the DUT [[202]] 200 via the test access port 208 and stored in the cache memory 204. The CPU core 202 accesses the assembled/compiled codes stored in the cache memory 204 and executes the code. As can be seen, execution of the compiled codes 212 results in an interaction between the CPU core 202 and the cache memory 204 as depicted by arrows A and B in Figure 2. This interaction does not involve any of the input/output (IO) mechanisms of the DUT 200, which are consequently not tested. Thus, the speed at which the ATE 210 can send

responses to the CPU core 202 is not an issue. This is one advantage of the testing methodology illustrated in Figure 2 over the testing methodology illustrated in Figure 1. However, the methodology illustrated with respect to Figure 2 suffers from a disadvantage in that the IO mechanisms are not tested. - -

Please replace Paragraph [0017] on Page 7 in the section entitled "Detailed Description" with the following paragraph:

- - **[0017]** The steps outlined in the flowchart of Figure 3 [[is]] <u>are</u> best understood with reference to Figure 5. - -

Please replace Paragraph [0017] on Page 7 in the section entitled "Detailed Description" with the following paragraph:

Please replace the paragraph on Page 21 under the heading entitled "Abstract" with the following paragraph:

- - A method <u>of testing a DUT</u> is provided. The method comprises loading a memory within a link-based system with a functional test program, executing the functional test program in a processor core of the link-based system, and routing test signals generated during execution of the functional test program to a response agent embedded in the link-based system via an external path. - -